

AMENDMENTS TO THE CLAIMS

Please amend the claims set forth below. Claims 1-35 are currently pending in the application. Claims 1, 3, 11, 18, 20, 30-35 have been amended without acquiescence to cited basis for rejection or prejudice to pursue in a related application. Claims 36-62 are new. A complete listing of the claims is provided below. No new matter is added.

1. (Currently Amended) A method for implementing efficient access to XML data, comprising:
 - receiving a schema for the XML data;
 - identifying an element within the schema to associate with a named access procedure;
 - determining if the element identified is appropriate for association with the named access procedure; and

if the element identified is appropriate for association, then creating the named access procedure and associating the named access procedure with the element, the named access procedure providing direct access to the element within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element without requiring progressive traversal of child nodes.

2. (Original) The method of claim 1 in which the named access procedure is defined based upon analysis of the schema.
3. (Currently Amended) The method of claim 2 in which an access parameters ~~are used to define the named access procedure, the access parameters including~~ includes offset information for the element.
4. (Original) The method of claim 1 in which the named access procedure is a procedure to get a value for the element or to set a value for the element.
5. (Original) The method of claim 1 in which the named access procedure performs a direct mapping to an intended datatype for the element.

6. (Original) The method of claim 5 in which a conversion to a string datatype is not performed when mapping to the intended datatype.
7. (Original) The method of claim 5 in which the mapping is to a close-matching datatype.
8. (Original) The method of claim 1 in which the element is not appropriate for association if it corresponds to a datatype of 'ANY' or is a node that is not defined in the schema.
9. (Original) The method of claim 1 in which the element is appropriate for association if it corresponds to a native datatype of the system in which the method is performed.
10. (Original) The method of claim 1 in which the named access procedure is implemented as a bean accessor type.
11. (Currently Amended) A method for performing efficient access to XML data, comprising:
 - identifying an element within an instance of the XML data to access;
 - determining if the element has been associated with a named access procedure corresponding to the element, the named access procedure providing direct access to the element within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element ~~without requiring progressive traversal of child nodes;~~
 - if the element has been associated with the named access procedure, then using the named access procedure to access the element in the instance of the XML data; and
 - if the element has not been associated with the named access procedure, then using a DOM API to access the element in the instance of the XML data.
12. (Original) The method of claim 11 in which a schema for the XML data is known apriori and the named access procedure is based upon analysis of the schema.

13. (Original) The method of claim 11 in which the named access procedure performs a direct mapping to an intended datatype for the element.
14. (Original) The method of claim 11 in which other elements of the data not presently needed are not loaded into memory.
15. (Original) The method of claim 11 in which the element is at a known offset from a parent location.
16. (Original) The method of claim 15 in which the mapping of the known offset is managed independently of the XML data.
17. (Original) The method of claim 15 in which a memory layout associated with the XML data is maintained as a flat layout.
18. (Currently Amended) A method for implementing efficient access to data that is based on a mark-up language, in which the data associated with a schema, the data comprising a parent node and at least one child node, the method comprising:
 - receiving the schema for the data that is based on the mark-up language;
 - identifying a child node that is to be accessed within the data;
 - reviewing the schema to determine one or more access parameters relating to the child node, wherein at least one access parameter is determined for the child node relative to the parent node; and
 - using the one or more access parameters to directly access the child node ~~without requiring progressive traversal of child nodes~~.
19. (Original) The method of claim 18 in which the mark-up language is based on XML, HTML, or SGML.

20. (Currently Amended) The method of claim 18 in which ~~the~~ at least one access parameters is based on offset position, data length, or datatype.
21. (Original) The method of claim 18 in which a named access procedure is defined to get a value for the child node or to set a value for the child node.
22. (Original) The method of claim 18 in which direct mapping is performed to an intended datatype for the child node.
23. (Original) The method of claim 18 in which the child node is not directly accessed if it corresponds to a datatype of 'ANY' or is a node that is not defined in the schema.
24. (Original) The method of claim 18 in which the child node is directly accessed if it corresponds to a native datatype of the system in which the method is performed.
25. (Original) The method of claim 18 in which direct access is performed to an offset location for the child node.
26. (Original) The method of claim 25 in which the child node is at a known offset from a location for the parent node.
27. (Original) The method of claim 26 in which a mapping of the known offset is managed independently of the data.
28. (Original) The method of claim 25 in which a memory layout associated with the data is maintained as a flat layout.
29. (Original) The method of claim 18 in which other child nodes not presently needed are not loaded into memory.
30. (Currently Amended) A system for implementing efficient access to XML data, comprising:
means for receiving a schema for the XML data;

means for identifying an element within the schema to associate with a named access procedure;

means for determining if the element identified is appropriate for association with the named access procedure; and

means for creating the named access procedure and associating the named access procedure with the element if the element identified is appropriate for association, the named access procedure providing direct access to the element within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element ~~without requiring progressive traversal of child nodes.~~

31. (Currently Amended) A computer program product comprising a computer usable medium having executable code to execute a process for implementing efficient access to XML data, the process comprising:

receiving a schema for the XML data;

identifying an element within the schema to associate with a named access procedure;

determining if the element identified is appropriate for association with the named access procedure; and

if the element identified is appropriate for association, then creating the named access procedure and associating the named access procedure with the element, the named access procedure providing direct access to the element within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element ~~without requiring progressive traversal of child nodes.~~

32. (Currently Amended) A system for performing efficient access to XML data, comprising:
means for identifying an element within an instance of the XML data to access;

means for determining if the element has been associated with a named access procedure corresponding to the element, the named access procedure providing direct access to the element

within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element without requiring progressive traversal of child nodes;

means for using the named access procedure to access the element in the instance of the XML data if the element has been associated with the named access procedure; and

means for using a DOM API to access the element in the instance of the XML data if the element has not been associated with the named access procedure.

33. (Currently Amended) A computer program product comprising a computer usable medium having executable code to execute a process for performing efficient access to XML data, the process comprising:

identifying an element within an instance of the XML data to access;

determining if the element has been associated with a named access procedure corresponding to the element, the named access procedure providing direct access to the element within an instance of the XML data with at least one access parameter that is determined for the element relative to a second element without requiring progressive traversal of child nodes;

if the element has been associated with the named access procedure, then using the named access procedure to access the element in the instance of the XML data; and

if the element has not been associated with the named access procedure, then using a DOM API to access the element in the instance of the XML data.

34. (Currently Amended) A system for implementing efficient access to data that is based on a mark-up language, in which the data associated with a schema, the data comprising a parent node and at least one child node, the method comprising:

means for receiving the schema for the data that is based on the mark-up language;

means for identifying a child node that is to be accessed within the data;

means for reviewing the schema to determine one or more access parameters relating to the child node, wherein at least one access parameter is determined for the child node relative to the parent node; and

means for using the one or more access parameters to directly access the child node ~~without requiring progressive traversal of child nodes.~~

35. (Currently Amended) A computer program product comprising a computer usable medium having executable code to execute a process for implementing efficient access to data that is based on a mark-up language, in which the data associated with a schema, the data comprising a parent node and at least one child node, the process comprising:

receiving the schema for the data that is based on the mark-up language;

identifying a child node that is to be accessed within the data;

reviewing the schema to determine one or more access parameters relating to the child node, wherein at least one access parameter is determined for the child node relative to the parent node; and

using the one or more access parameters to directly access the child node ~~without requiring progressive traversal of child nodes.~~

36. (New) The method of claim 18 in which direct access is performed to a location in an XML document for the child node.

37. (New) The system of claim 30 in which the named access procedure is defined based upon analysis of the schema.

38. (New) The system of claim 30 in which an access parameter includes offset information for the element.

39. (New) The system of claim 30 in which the named access procedure performs a direct mapping to an intended datatype for the element.
40. (New) The system of claim 30 in which the element is not appropriate for association if it corresponds to a datatype of 'ANY' or is a node that is not defined in the schema.
41. (New) The system of claim 30 in which the element is appropriate for association if it corresponds to a native datatype of the system.
42. (New) The computer program product of claim 31 in which the named access procedure is defined based upon analysis of the schema.
43. (New) The computer program product of claim 31 in which an access parameter includes offset information for the element.
44. (New) The computer program product of claim 31 in which the named access procedure performs a direct mapping to an intended datatype for the element.
45. (New) The computer program product of claim 31 in which the element is not appropriate for association if it corresponds to a datatype of 'ANY' or is a node that is not defined in the schema.
46. (New) The computer program product of claim 31 in which the element is appropriate for association if it corresponds to a native datatype of the system.
47. (New) The system of claim 34 in which the mark-up language is based on XML, HTML, or SGML.
48. (New) The system of claim 34 in which at least one access parameter is based on offset position, data length, or datatype.

49. (New) The system of claim 34 in which direct mapping is performed to an intended datatype for the child node.
50. (New) The system of claim 34 in which the child node is directly accessed if it corresponds to a native datatype of the system in which the method is performed.
51. (New) The computer program product of claim 35 in which the mark-up language is based on XML, HTML, or SGML.
52. (New) The computer program product of claim 35 in which at least one access parameter is based on offset position, data length, or datatype.
53. (New) The computer program product of claim 35 in which direct mapping is performed to an intended datatype for the child node.
54. (New) The computer program product of claim 35 in which the child node is directly accessed if it corresponds to a native datatype of the system in which the method is performed.
55. (New) The system of claim 32 in which the named access procedure performs a direct mapping to an intended datatype for the element.
56. (New) The system of claim 32 in which other elements of the data not presently needed are not loaded into memory.
57. (New) The system of claim 32 in which the element is at a known offset from a parent location.
58. (New) The system of claim 32 in which a memory layout associated with the XML data is maintained as a flat layout.
59. (New) The computer program product of claim 33 in which the named access procedure performs a direct mapping to an intended datatype for the element.

60. (New) The computer program product of claim 33 in which other elements of the data not presently needed are not loaded into memory.
61. (New) The computer program product of claim 33 in which the element is at a known offset from a parent location.
62. (New) The computer program product of claim 33 in which a memory layout associated with the XML data is maintained as a flat layout.